AD-A043 017 AIR FORCE HUMAN RESOURCES LAB BROOKS AFB TEX F/G 9/2
COMPUTER GRAPHICS: TWO- AND THREE-DIMENSIONAL CLIPPING.(U)
MAY 77 M L CYRUS, J BECK

UNCLASSIFIED AFHRL-TR-77-14 NL

| OF |
AD
A043017

END
DATE
FILMED
9-77
DDC

AFHRL-TR-77-14

AIR FORCE HUMAN RESOURCES LABORATORY

COMPUTER GRAPHICS:
TWO- AND THREE-DIMENSIONAL CLIPPING

By
Michael L. Cyrus

FLYING TRAINING DIVISION
Williams Air Force Base, Arizona 85224

Jay Beck
Tektronix, Incorporated
P.O. Box 500
Beaverton, Oregon 97005

May 1977

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235

NOTICE

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This interim report was submitted by Flying Training Division, Air Force Human Resources Laboratory, Williams Air Force Base, Arizona 85224, under project 1123, with HQ Air Force Human Resources Laboratory (AFSC), Brooks Air Force Base, Texas 78235.

This report has been reviewed and cleared for open publication and/or public release by the appropriate Office of Information (OI) in accordance with AFR 190-17 and DoDD 5230.9. There is no objection to unlimited distribution of this report to the public at large, or by DDC to the National Technical Information Service (NTIS).

This technical report has been reviewed and is approved for publication.

EDWARD E. EDDOWES, Technical Adviser
Flying Training Division

DAN D. FULGHAM, Colonel, USAF
Commander

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFHRL-TR-77-14
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): COMPUTER GRAPHICS: TWO- AND THREE-DIMENSIONAL CLIPPING.
5. TYPE OF REPORT & PERIOD COVERED: Working Paper; Interim
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Michael L. Cyrus; Jay Beck
8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Flying Training Division
Air Force Human Resources Laboratory
Williams Air Force Base, Arizona 85224

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
62703F
11230301
17 03

11. CONTROLLING OFFICE NAME AND ADDRESS
HQ Air Force Human Resources Laboratory (AFSC)
Brooks Air Force Base, Texas 78235

12. REPORT DATE: May 1977
13. NUMBER OF PAGES: 18

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*
12 18 p.

15. SECURITY CLASS. *(of this report)*: Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*
Approved for public release; distribution unlimited.

9 Interim rept.,

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

clipping algorithms
computer graphics
visual displays

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

A clipping algorithm is derived and both two- and three-dimensional implementations of it are discussed. The algorithm finds the proper intersection of a line with any convex planar, polygon or spacial polyhedron. By interpretation of the computed clipping coefficients, both interior and exterior clipping to a convex region can result. Finally, the implementation lends itself well to highly parallel execution which reduces execution time.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified 404 415

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

COMPUTER GRAPHICS: TWO- AND THREE-DIMENSIONAL CLIPPING

I. INTRODUCTION

Clipping is a procedure used to determine the proper line segment(s) of a given line, such that the resulting segment(s) contain no points exterior to a plane that serves to define a boundary. Clipping is employed to limit the amount of work a display device must perform by drawing only the visible parts of a line. This is done by using a clipping algorithm to compute only the line segments interior (or exterior) to a defined region that has been selected for display. The resulting list of lines represents a substantially smaller amount of work for the display hardware to process in a given period of time.

A good clipping algorithm quickly rejects lines that lie outside the viewing area, and since speed is always desirable, algorithms that lend themselves to parallel (or pipeline implementation), in either hardware or firmware, are sought.

Let us initially limit our discussion to two-dimensional clipping. A later section of this paper will explore three-dimensional clipping.

II. TWO-DIMENSIONAL CLIPPING

Typically, display regions have been chosen as being rectangular (Figure 1). The actual number of sides is not of importance to the development of a clipping algorithm, although it does affect performance when finally implemented. Viewing windows are always chosen however, as convex sets in a Hilbert Space.

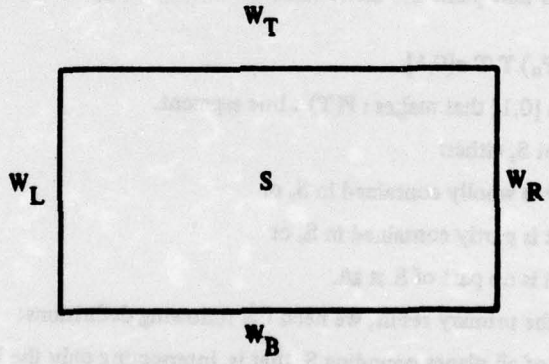


Figure 1. Rectangular display region.

The standard properties of Hilbert spaces, as well as the set concepts referred to in this paper, are found in most texts on linear analysis.

Making use of these properties, the following is clear: A straight line, intersecting the interior of a convex set, can intersect the boundary of the set in, at most, two places. In the event the convex set is closed and bounded, the straight line will intersect it in exactly two places.

At this point, we introduce the idea of "inward" normals. That is, we say n is an inward normal for any boundary point X of S if, for any other point Y in S,

$n \cdot (Y-X) > 0$. where "$\cdot$" denotes the inner product operation.

With this representation, we have defined a natural geometry over the set S.

For the set S only four unique inward normals exist. All others are mathematically equivalent to these four.

Our particular clipping problem, that of determining the points of intersection between a particular line segment and the set S can then be partially resolved by examining the parameterized behavior of the

line segment with the four inward normals to S. In particular, if f is a boundary point of S for which n is an inward vector and P(T) is a parameterization of the line segment in question, then, for particular values of T,

$$n \cdot [P(T) - f] > 0$$

implies P(T) is "pointed" toward the interior of S with respect to at least one of the four normals,

$$n \cdot [P(T) - f] = 0$$

implies P(T) is "pointed" parallel to the plane containing f and perpendicular to n (that is, parallel to this particular boundary of S) and

$$n \cdot [P(T) - f] < 0$$

implies P(T) is "pointed" away from the set S.

So far we have mentioned only that any line piercing S does so at exactly two points. It is also clear that these two points of intersection do not lie on the same target planes to the boundary of S (a plane not completely containing a line may be intersected by it at most once). These facts give rise to our first important result:

if f is a point in a particular plane bounding S for which n is an inward normal, then the scalar equation

$$n \cdot [P(T) - f] = 0$$

has, at most, one solution.

Thus, there are exactly two possibilities; first, P(T) is parallel to some plane lying along the boundary of S. Second P(T) pierces that plane for some value of T. Now, suppose we want to talk about drawing only a line segment:

$$: P(T) = P_0 + (P_1 - P_0)\,T \; T \in [0,1]$$

It is the restriction of T to [0,1] that makes : P(T) a line segment.

With regard to the set S, either:

1. The line segment is wholly contained in S, or
2. The line segment is partly contained in S, or
3. The line segment is no part of S at all.

Before introducing the primary result, we need the following definitions:

Define S_b as the set of all planes bounding S, that is, intersecting only the boundary of S.

For each $b \in S_b$, define n(b) as the set of positive normals drawn from b with respect to S.

The Primary Result

The set of all T such that:

1. $T \in (0,1)$
2. $n \cdot [P(T) - f] > 0$

Where for every $b \in S_b$, $n \in n(b)$, and f is an arbitrary vector in b, completely determines the line segment containment in S.

Careful examination of the flowchart (Figure 2) reveals the underlying geometry involved in this technique. The implementation lends itself to swift line segment rejection.

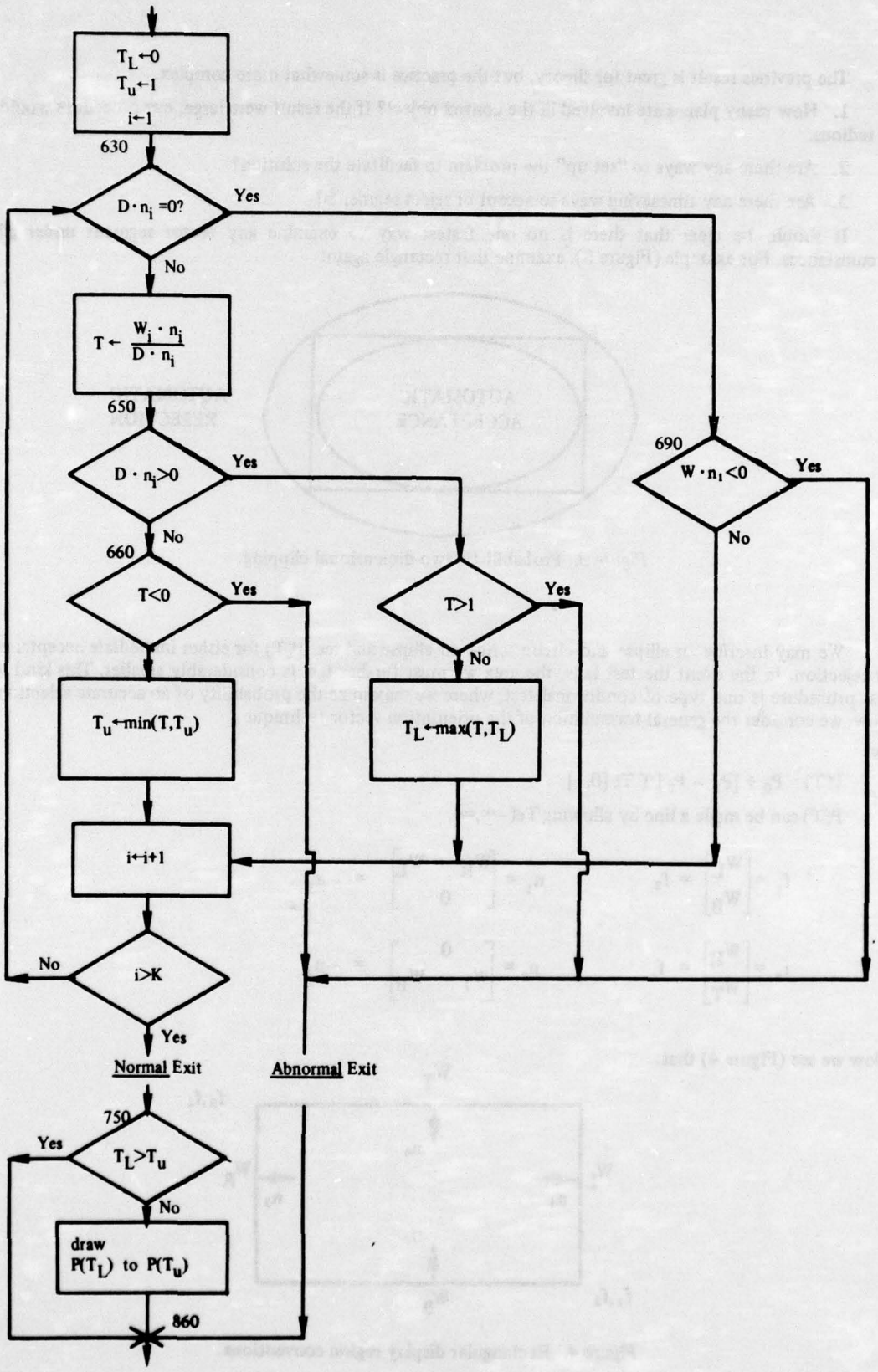


Figure 2. **Primary climbing algorithm flowchart.**

The previous result is great for theory, but the practice is somewhat more complex.

1. How many planes are involved in the convex object? If the result were large, our procedure would be tedious.

2. Are there any ways to "set up" the problem to facilitate the solution?

3. Are there any timesaving ways to accept or reject segments?

It should be clear that there is no one fastest way to examine any vector segment under all circumstances. For example (Figure 3), examine that rectangle again:

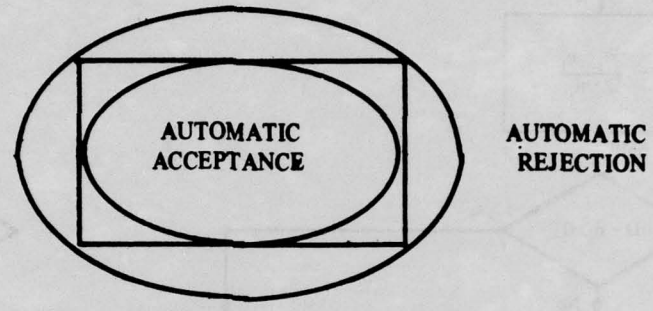


Figure 3. **Probabilistic two-dimensional clipping.**

We inscribe an ellipse and circumscribe an ellipse and test P(T) for either immediate acceptance or rejection. In the event the test fails, the area we must further test is considerably smaller. This kind of test procedure is one type of conditional test, where we maximize the probability of an accurate selection. Now, we consider the general formulation of the orientation vector technique.

Let

$$P(T) = P_0 + [P_1 - P_0]\,T \quad T \epsilon [0,1]$$

P(T) can be made a line by allowing $T \epsilon (-\infty, \infty)$.

$$f_1 = \begin{bmatrix} W_L \\ W_B \end{bmatrix} = f_2 \qquad n_1 = \begin{bmatrix} W_R - W_L \\ 0 \end{bmatrix} = -n_3$$

$$f_3 = \begin{bmatrix} W_R \\ W_T \end{bmatrix} = f_4 \qquad n_2 = \begin{bmatrix} 0 \\ W_T - W_B \end{bmatrix} = -n_4$$

Now we see (Figure 4) that:

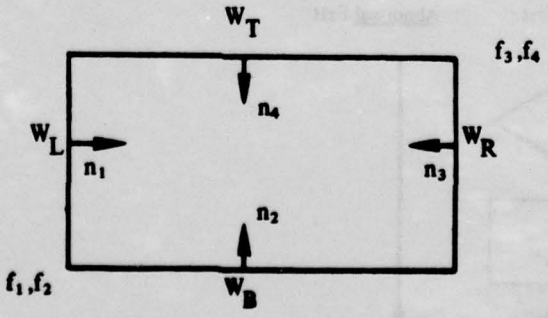


Figure 4. **Rectangular display region conventions.**

The five conditions become

$$[P(T) - f_i] \cdot n_i > 0 \qquad i = 1, 2, 3, 4$$

$$\text{or } P_0 \cdot n_i + T[P_1 - P_0] \cdot n_i > 0 \qquad i = 1, 2, 3, 4$$

$$\text{or } T[P_1 - P_0] \cdot n_i > [f_i - P_0] \cdot n_i \qquad i = 1, 2, 3, 4$$

and $T \epsilon [0, 1]$

letting

$P_1 - P_0 = D$ (D is called the Directorix of the vector P(T))

and $W_i = P_0 - f_i$ (W_i is the ith plane "weight" vector), $\qquad i = 1, 2, 3, 4$

Then we can write:

$$TD\, n_i + W_i \cdot n_i > 0 \qquad i = 1, 2, 3, 4$$

Now we have our algorithm:

For a K-sided convex object S with inside normals:

$$n_1, n_2, \ldots, n_k$$

Points on each plane:

$$f_1, f_2 \ldots, f_k$$

Let P(T) be the parameterization of the line running from P_0 to P_1

Define:

$$W_i = P_0 - f_i, D = P_1 - P_0$$

Then the K + 1 conditions for any portion of the line segment being in S are:

$$TD \cdot n_i + W_i \cdot n_i > 0 \qquad i = 1, 2, \ldots, K$$

$$T \epsilon [0, 1]$$

Let the set of such that each ith case above is satisfied:

$$T_i = \{ T \mid T\, D \cdot n_i + W_i \cdot n_i > 0 \}$$

Then for each T_i, $T_i \epsilon (-\infty, \infty)$

Let $T = \bigwedge_{i=1}^{k} T_i$

Then $T = \{ T \mid P(T) \epsilon S \}$

Finally, $T^* = T \cap [0, 1]$

Using the Tektronix 4051 Graphic Computing System, an implementation of this algorithm was programmed (Figure 5). The program and sample results follows (Figure 6):

In this program:

$Po_x = X1 \qquad P1_x = X2$

$Po_y = Y1 \qquad P1_y = Y2$

Figures 7 and 8 show examples of segments drawn and rejected, respectively.

We can introduce efficiency into the computation by first evaluating directly for the values of T.

```
100  INIT
110  PAGE
120  WINDOW 0,22,0,22
130  VIEWPORT 65,130,0,65
140  DIM F(4,2),N(4,2)
150  REM ..... Define window
160  W1=7
170  W2=18
180  W3=10
190  W4=17
200  REM ..... Draw axis
210  MOVE 0,0
220  DRAW 22,0
230  MOVE 0,0
240  DRAW 0,22
250  REM ..... Draw Window boundary
260  MOVE W1,W3
270  DRAW W2,W3
280  DRAW W2,W4
290  DRAW W1,W4
300  DRAW W1,W3
310  REM ..... Enter line definition
320  PRINT @32,21,0,90
330  PRINT"Enter x1,y1, x2,y2 :";
340  INPUT X1,Y1,X2,Y2
350  REM ..... Normals defined
360  N(1,1)=W2-W1
370  N(1,2)=0
380  N(3,1)=-N(1,1)
390  N(3,2)=-N(1,2)
400  N(2,1)=0
410  N(2,2)=W4-W3
420  N(4,1)=-N(2,1)
430  N(4,2)=-N(2,2)
440  REM ..... Window boundary
450  F(1,1)=W1
460  F(1,2)=W3
470  F(2,1)=F(1,1)
480  F(2,2)=F(1,2)
490  F(3,1)=W2
500  F(3,2)=W4
510  F(4,1)=F(3,1)
520  F(4,2)=F(3,2)
530  REM ..... Algorithm
540  D1=X2-X1
550  D2=Y2-Y1
560  L=0
570  U=1
580  FOR I=1 TO 4
590  W1=X1-F(I,1)
600  W2=Y1-F(I,2)
610  D=D1*N(I,1)+D2*N(I,2)
620  W=W1*N(I,1)+W2*N(I,2)
630  IF D=0 THEN 690
640  T=-W/D
650  IF D>0 THEN 710
660  IF T<0 THEN 860
670  U=T MIN U
680  GO TO 730
690  IF W<0 THEN 680
700  GO TO 730
710  IF T>1 THEN 860
720  L=T MAX L
730  NEXT I
740  REM ..... Segment drawn
750  IF L=>U THEN 860
760  X4=X1+(X2-X1)*I
770  Y4=Y1+(Y2-Y1)*I
780  X5=X1+(X2-X1)*U
790  Y5=Y1+(Y2-Y1)*U
800  MOVE X4,Y4
810  DRAW X5,Y5
820  PRINT @32,21:0,30
830  PRINT "SEGMENT DRAWN"
840  END
850  REM ..... Segment rejected
860  PRINT @32,21:0,30
870  PRINT "SEGMENT REJECTED"
880  END
```

Figure 5. **Sample computer program.**

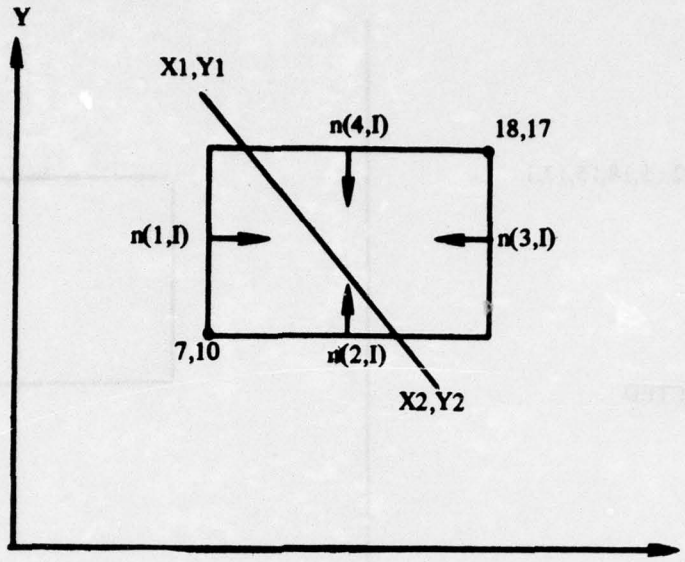


Figure 6. **Sample display problem.**

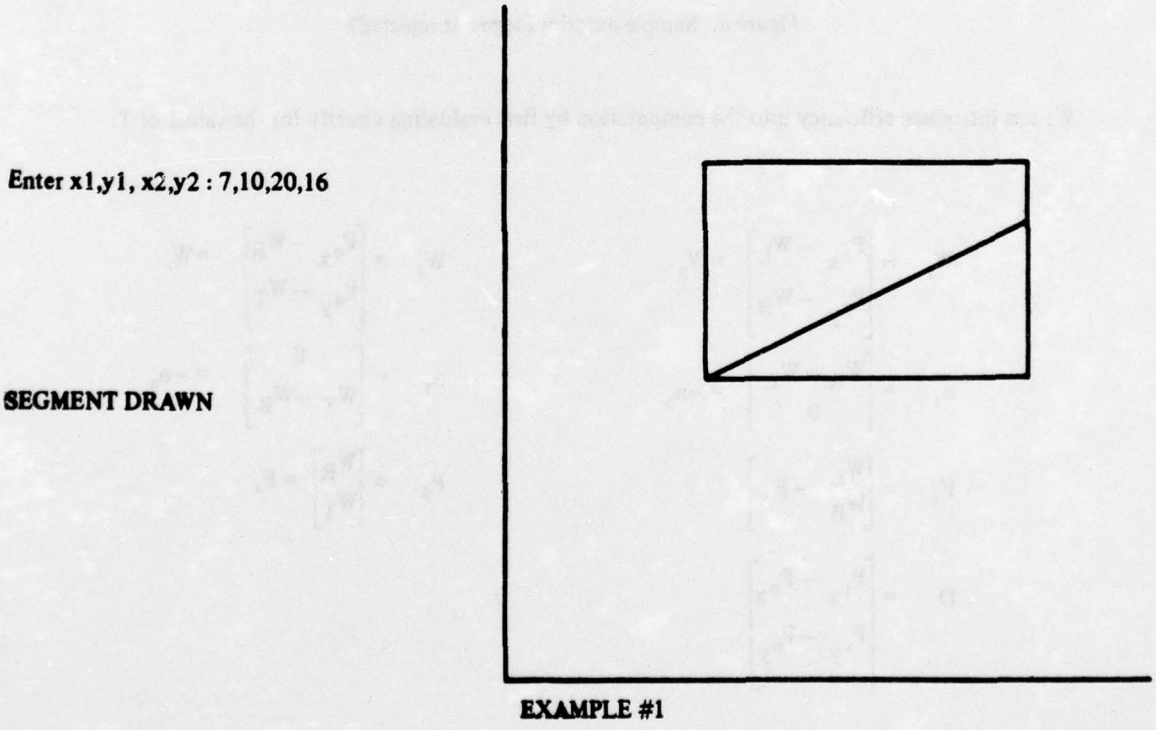
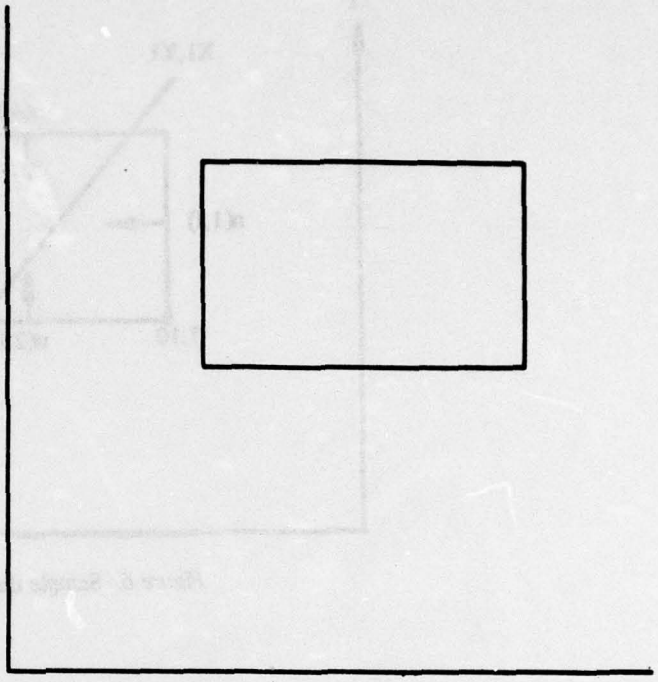


Figure 7. **Sample interior (segment drawn)**

Enter x1,y1, x2,y2 : 5,18,15,17.1

SEGMENT REJECTED



EXAMPLE #2

Figure 8. **Sample exterior (segment rejected).**

We can introduce efficiency into the computation by first evaluating directly for the values of T.

$$W_1 = \begin{bmatrix} P_{0_x} - W_L \\ P_{0_y} - W_B \end{bmatrix} = W_2 \qquad W_3 = \begin{bmatrix} P_{0_x} - W_R \\ P_{0_y} - W_T \end{bmatrix} = W_4$$

$$n_1 = \begin{bmatrix} W_R - W_L \\ 0 \end{bmatrix} = -n_3 \qquad n_2 = \begin{bmatrix} 0 \\ W_T - W_B \end{bmatrix} = -n_4$$

$$F_1 = \begin{bmatrix} W_L \\ W_B \end{bmatrix} = F_2 \qquad F_3 = \begin{bmatrix} W_R \\ W_T \end{bmatrix} = F_4$$

$$D = \begin{bmatrix} P_{1_x} - P_{0_x} \\ P_{1_y} - P_{0_y} \end{bmatrix}$$

(1) $T(P_{1_X} - P_{0_X})(W_R - W_L) > (W_L - P_{0_X})(W_R - W_L)$

(2) $T(P_{1_Y} - P_{0_Y})(W_T - W_B) > (W_B - P_{0_Y})(W_T - W_B)$

(3) $T(P_{1_X} - P_{0_X})(W_L - W_R) > (W_R - P_{0_X})(W_L - W_R)$

(4) $T(P_{1_Y} - P_{0_Y})(W_R - W_T) > (W_T - P_{0_Y})(W_B - W_T)$

(5) $T \in [0,1]$

Assuming that P(T) is not parallel to any side of the rectangle, then we can solve for the values of T so that each equation (1 through 4) is satisfied identically.

$$(1) \quad = \quad \frac{W_L - P_{0_X}}{P_{1_X} - P_{0_X}}$$

$$(2) \quad = \quad \frac{W_B - P_{0_Y}}{P_{1_Y} - P_{0_Y}}$$

$$(3) \quad = \quad \frac{W_R - P_{0_X}}{P_{1_X} - P_{0_Y}}$$

$$(4) \quad = \quad \frac{W_T - P_{0_Y}}{P_{1_Y} - P_{1_Y}}$$

III. THREE-DIMENSIONAL CLIPPING

Since the algorithm was not restricted to two dimensions, the basic result applies in the three-dimensional case. We will examine a particular case. The set of points chosen for perspective purposes is centered with the axis center at the eye point, looking along the Z axis. The set of points is given by the set product

$$\{W_L, W_R\} \times \{W_B, W_T\} \times \{W_H, W_Y\}$$

The region in space R, now being viewed as our convex set, is given by the product

$$[W_L, W_R] \times [W_B, W_T] \times [W_H, W_Y]$$

In the 2-D case, there were four equations in T and one additional constraint. In this 3-D case, there are six equations in T and that same constraint.

There are six bounding planes, four of which contain the point [0,0,0]. The other two are the hither and yon planes, which are handled as special cases (Figure 9). Symmetry conditions imply the following:

$$W_L = -W_R \qquad W_B = -W_T$$

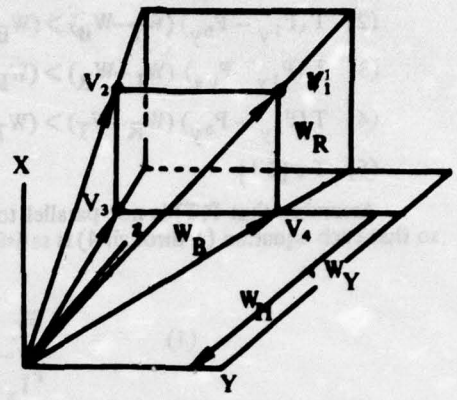


Figure 9. **Three-dimensional viewing window.**

Let

$V_1 = [W_R, W_T, W_H]$

$V_2 = [-W_R, +W_T, W_H]$

$V_3 = [-W_R, -W_T, W_H]$

$V_4 = [+W_R, -W_T, W_H]$

These four vectors are the vectors along the intersection of the four bounding planes to the first perspective plane.

$n_1 = V1 \times V_2 = [0, -2W_RW_H, 2W_RW_T]$

$n_2 = V_2 \times V_3 = [2W_TW_H, 0, 2W_RW_T]$

$n_3 = V_3 \times V_4 = [0, 2W_RW_H, 2W_RW_T]$

$n_4 = V_4 \times V_1 = [-2W_TW_H, 0, 2W_RW_T]$

$n_5 = (\text{Hither}) = [\,0, 0, 1\,]$

$n_6 = (\text{Yon}) = [\,0, 0, -1\,]$

$$f_1 = \begin{bmatrix}0\\0\\0\end{bmatrix} = f_2 = f_3 = f_4$$

$$f_5 = \begin{bmatrix}0\\0\\W_H\end{bmatrix}$$

$$f_6 = \begin{bmatrix}0\\0\\W_Y\end{bmatrix}$$

$D = P_1 - P_0$

$W_L = -f_1 + P_0$

$W_1 = +P_0 \qquad W_2 = +P_0 \qquad W_3 = +P_0 \qquad W_4 = +P_0$

$$W_5 = \begin{bmatrix}+P_{0_X}\\+P_{0_y}\\-W_H+P_{0_Z}\end{bmatrix} \qquad W_6 = \begin{bmatrix}+P_{0_X}\\+P_{0_y}\\-W_y+P_{0_Z}\end{bmatrix}$$

The same rules apply to $D \cdot n_i$ and $w_i \cdot n_i$ as before.

The primary formula is $T = \frac{-w_i \cdot n_i}{D \cdot n_i}$ for each i, giving

$$T_1 = \frac{W_T P_{0_z} + W_H P_{0_y}}{(P_{1_z} - P_{0_z})\ W_T - (P_{1_y} - P_{0_y})\ W_H}$$

$$T_2 = \frac{W_H P_{0_x} + W_R P_{0_z}}{(P_{1_x} - P_{0_x})\ W_H + (P_{1_z} - P_{0_z})\ W_R}$$

$$T_3 = - \frac{W_H P_{0_y} + W_T P_{0_z}}{(P_{1_y} - P_{0_y}) W_H + (P_{1_z} - P_{0_z}) W_T}$$

$$T_4 = \frac{-W_R P_{0_z} + W_H P_{0_x}}{(P_{1_z} - P_{0_z}) W_R - (P_{1_x} - P_{0_x}) W_H}$$

$$T_5 = \frac{-P_{0_z} + W_H}{P_{1_z} - P_{0_z}}$$

$$T_6 = \frac{-W_y + P_{0_z}}{P_{0_z} - P_{1_z}}$$

These values of T, define the intersection points of $P(T)$ with each bounding plane.

It can be seen from this analysis that while the computation of T for the 3-D case may appear to be a lengthy computation, each value of T can be computed in a separate parallel process.

Our discussion so far has been limited to determining a line segment that is either rejected or clipped to fit inside a convex set S. Our output (T_L, T_U) can also be used to draw the complement, that is the line segment(s) exterior to a given convex set S.

The possibilities are summarized in Figure 10.

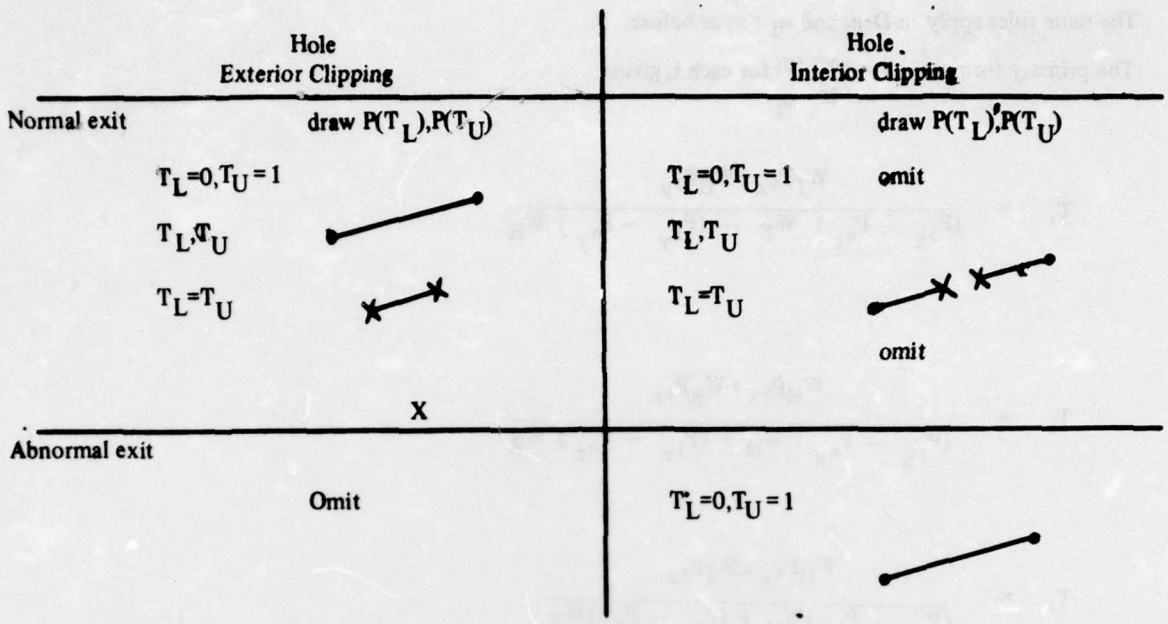


Figure 10. **Window-hole clipping summary.**

This introduces the possibility that an Algol, PL/1 or Pascal implementation of this algorithm might be implemented to first apply exterior clipping to a 'Hole' contained within the convex window.

If we define a function which performs P(T) cal. EG, the flow chart (Figure 2) previously drawn can be extended as in Figure 11. The output of SE . scaled appropriately for a display device using integer coordinates.

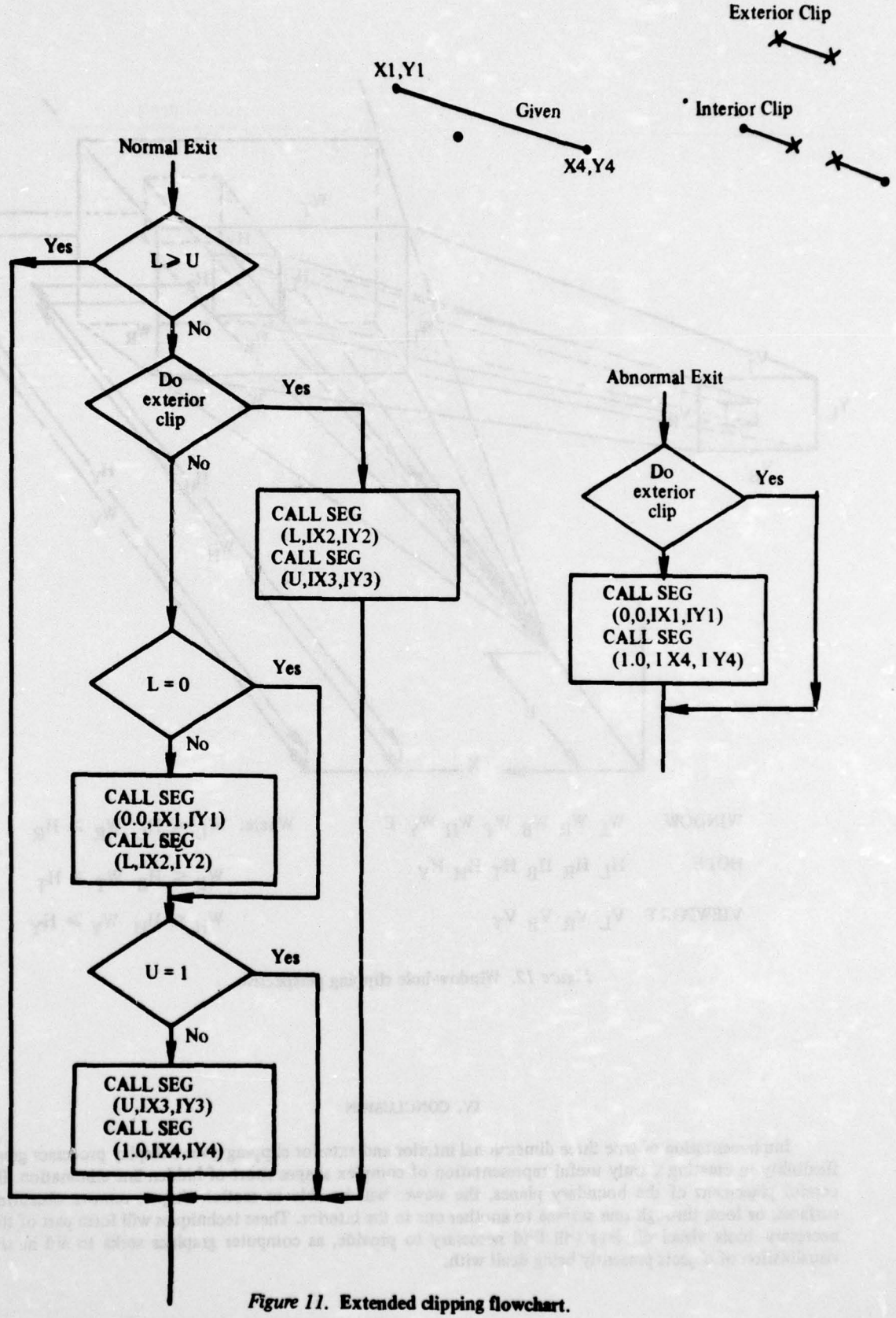
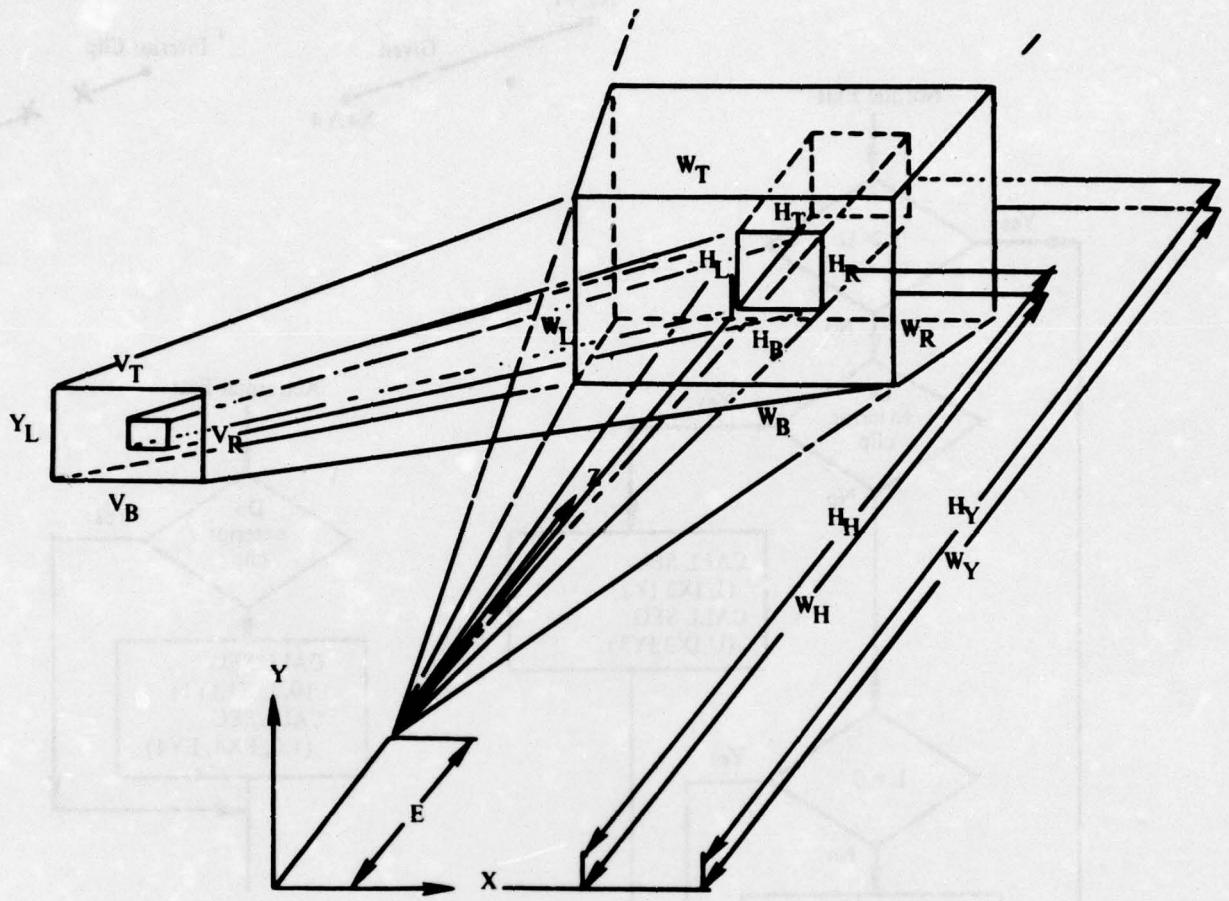


Figure 11. **Extended clipping flowchart.**



WINDOW	$W_L\ W_R\ W_B\ W_T\ W_H\ W_Y\ E$	Where: $W_L \leq H_L\ W_R \geq H_R$
HOLE	$H_L\ H_R\ H_B\ H_T\ H_H\ H_Y$	$W_B \leq H_B\ W_T \geq H_T$
VIEWPORT	$V_L\ V_R\ V_B\ V_T$	$W_H \leq H_H\ W_Y \geq H_Y$

Figure 12. **Window-hole clipping perspective.**

IV. CONCLUSION

Implementation of true three dimensional interior and exterior clipping gives a display processor great flexibility in creating a truly useful representation of complex shapes short of hidden line elimination. By careful placement of the boundary planes, the viewer will be able to section shapes, remove obscuring surfaces, or look through one surface to another one in the interior. These techniques will form part of the necessary tools visual displays will find necessary to provide, as computer graphics seeks to aid in the visualization of objects presently being dealt with.

☆ U.S. GOVERNMENT PRINTING OFFICE: 1977—771-057/46